Digital Control Pulse Delays

What goes up must come down!

July 12, 1996

Introduction

One might consider that digital control in an accelerator control system consists merely of setting a bit to a "0" or a "1." In hardware terms, this is likely to be what is provided. But when the hardware which is being controlled is taken into account, the situation is a bit more complex.

Turning a power supply on, for example, may be accommodated by a single control line; but often, the supply expects to be driven by pulses rather than a simple level set hi or lo. Whereas a single pulse might drive a reset line to a power supply, it normally takes two control lines to support pulsed on-off control. And, to make it interesting, the on-off *status* of the supply is normally provided as a single status bit.

Timing is important also. The pulses acceptable for control of a large power supply may have to be asserted for a significant fraction of a second, in order for associated relay logic to make up. The required length of the pulse can vary depending on the hardware being controlled.

A facility for generating digital pulses from *memory-mapped* digital output lines has always been part of the VME system software. It was used in the earlier Linac control system for building pulse control of power supply on-off commands, where the length of the pulse might have to be about 0.5 seconds in order to allow for heavy duty relays to make up.

This note concerns extension of the system so that *non-memory-mapped* hardware can also be driven to form pulses—especially for 1553 hardware.

New pulse logic

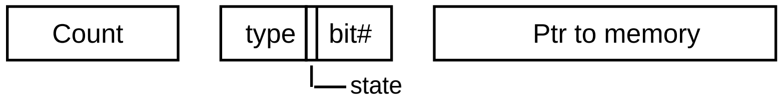
Since the present memory-mapped pulse-forming logic is part of the 15 Hz interrupt routine, it must be moved to the task level, where 1553 I/O can be handled. A reasonable place to put this logic is in the Update Task, which is concerned with processing entries in the Data Access Table to update the data pool. It could be placed either before or after reading the data. If it is placed *after* reading the data, then there will be fresh readings of the 1553 digital control data to use as a basis for setting or clearing a particular bit.

Hardware/software control of pulse delays

The timing of a pulse can be done either in hardware or software. If it is done in hardware, the software must be aware of it, so it will not then try to reset the bit to its non-asserted state after the time delay. And, of course, if it is done in passed.

DCTABLE format

The Digital Control Table is used by the system to support digital pulse delays. It consists of a set of 8-byte entries in the following format:



The count word times out the delay in 15 Hz cycles. The type byte contains the entry type. The entry types used are:

0: *memory-mapped digital control*

In this case, the bit# is a value in the range 0–7, and the Ptr is the address of a byte of memory. When the count reaches zero, the designated bit of memory is either set or cleared according to the value of the state bit.

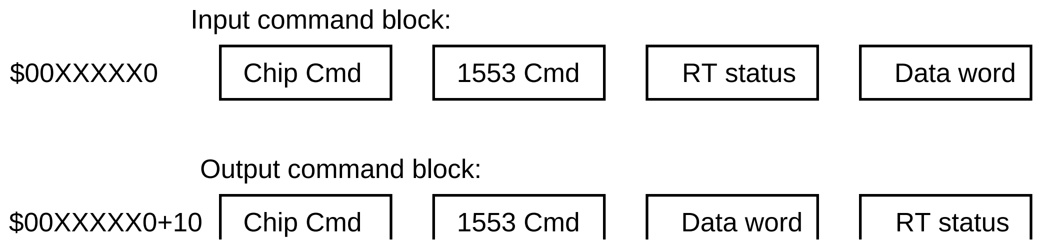
1: *1553 digital control*

Here, the bit# is a value in the range 0–15, and the Ptr is the address of the data word in the input command block in the 1553 controller's memory. (The input data word is simply the readback of the output data word.) When the count reaches zero, the word of input data is retrieved, the desig nated bit is set or cleared, and the word is output to the 1553 interface.

1553 digital control—background

The interface to hardware connected to 1553 remote terminals is handled by the VME system software through command block data structures. These blocks are located in the 1553 controller's own non-volatile memory. Each controller can house up to 64K of memory, and the `COM1553B` chip that actually drives the 1 MHz serial protocol reads and writes this memory via DMA. The command block structure houses the 1553 command word, which contains the addressing information for the RT (Remote Terminal interface to a D0 rack monitor, for example) and provides space for the chip to place a status word and up to 32 words of data it can receive from an RT. For output it contains the data word to be sent to the RT and the status response word.

Control of 1553 digital hardware requires two command blocks as follows:



command block by 16 bytes. So, if the pointer to the input data word is known, the location of the output command block can be determined.

The input data word is sampled, the control bit is set or cleared, and the data word is stored in the output command block and sent to the hardware. The input data word is then updated to match the word which was sent out, in order that subsequent control actions to other bits in the same word can be handled before another reading is made of the input data value. Note that in the case where *hardware* control of a pulse delay is used, this update must *not* be performed, lest subsequent pulse action of another bit in the same word cause a repeat of the first bit's pulse; furthermore, the hardware readback of the control lines must deliver the non-asserted state for any reading of a hardware controlled pulse bit for the same reason.

Software interface for digital control

Two means are provided for delivering settings which result in digital control actions. The first is closer to the hardware; the second is another step away from it.

The first type uses listype #21 to do digital I/O. In this case, the ident used is a Bit#. The 4-byte format is similar to that used for analog channels, but the index value is a Bit# instead of a channel#. The two ident formats are:

lan	node
Bit#	

lan	node
Chan#	

The format of the *data* for Bit-style digital control is two bytes in the form:

type	delay

The digital control type values are:

0:	None (no control)
1:	Toggle bit
2:	Set bit hi (to 1)
3:	Set bit lo (to 0)
4:	Pulse bit hi for delay time
5:	Pulse bit lo for delay time
6:	Pulse one of a pair of bits hi for delay time
7:	Pulse one of a pair of bits lo for delay time
8–11:	Not used
12–15:	Same as 4–7 but hardware-controlled

The delay byte gives the desired pulse delay in 15 Hz cycle units. A value of zero means a “short” pulse. For memory-mapped hardware, this is intended to be about 20 µ sec; for 1553 hardware, it will be much longer than that due to much larger software overhead in the 1553 driver. If the delay has the value of 1, the actual delay time will be less than one 15 Hz cycle, depending upon when in the 15 Hz cycle the command was issued. If a delay of many msec is required, one should probably use at least a value of 2 to insure a delay of at least one 15 Hz cycle. In the case that non-pulsed control is desired, using types 1–3, one may optionally send only the single byte of data for the setting, as the delay byte is not needed.

The second type of command resulting in digital control is the Chan-style digital control. This is done via listype#22, using the channel form of ident. In this case, there is more going on behind the scene. The channel-indexed descriptor entry contains fields which relate both to that channel’s associated digital status and digital control. *This is not simple, so bear with me.*

The form of data in the setting command for chan-style digital control is a two byte value as follows:

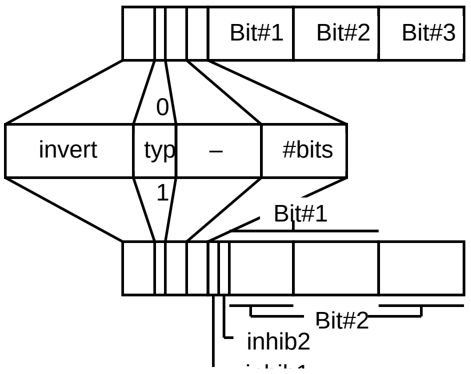
ctrl bit	state

The ctrl bit has a value which designates which of the digital control bits is to be used of those associated with that channel. In the current system, the value of this byte has a valid range of 1–3. The state refers to which of two alternative control actions are to be performed on that bit. Only the least significant bit in the byte is used for this value.

The format of the digital status and digital control fields must both be examined to see what choices there are for these digital control actions.

ADESC digital status field

The format of the digital status field has two variations:



first of the four bytes is shown in expanded view for clarity. The invert bits are used by the system software to invert the sense of the raw data that is returned in response to a data request for channel-associated status using listype#5.

There are 3 invert bits which correspond to the possible 3 status bits. These are sampled from most to least significant—that is, left-to-right—in order of successive Bit#'s 1,2 and 3. The least significant pair of bits gives the number of bits related to channel-associated digital status and control for this channel.

Bit#4 of this first byte is used to identify which format type is in use. For format type#0, up to 3 status bits can be supported which are associated with a given channel. The restriction is that they must be Bit#'s in the range 0-255. The parameter page on the small consoles recognizes the value $FF as denoting an inhibit of display of status data for that bit. (It may be used for control only.) This is used for control actions which have no related status bit.

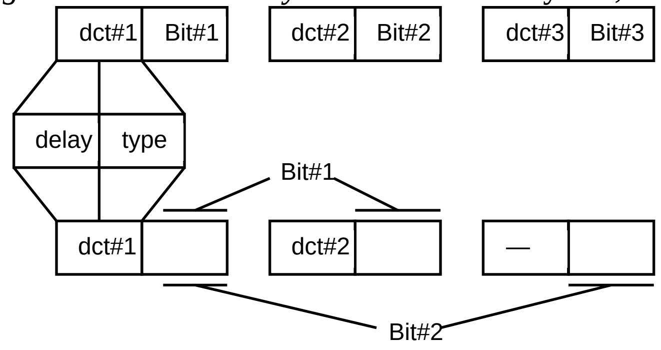
For format type#1, either one or two status bits can be supported. The first Bit# is given by the next two bytes; a second Bit# can be declared using the 2nd and 4th bytes of the 4-byte field. The obvious restriction is that the second Bit# must be in the same 256-bit block of Bit#'s as the first Bit#. In practice, this tends not to be a significant limitation. For this case, the parameter page logic samples the top two bits to get inhibit information about its status display for these bits.

Text for status display

The text used by the parameter page for display of digital status data is found as part of the title text for that channel. When there is a single Bit# used, the last 6 characters of the title (characters 13–18) denote two states—the first 3 characters for the nominal "0" state and the last 3 for the nominal "1" state. (The invert bits are used to reverse this logic from the nominal.) When 2 or 3 bits are used under format type #0, the same 6 characters are used to indicate both states of each bit; therefore, only a single character is available for each state. When 2 bits are used under format type#1, a second 6-character field is taken from the title for support of display of the second bit's status. This field is characters 6–11 of the 18-character title field. In this case, there is not much room left for any descrip tion of the channel, so the channel's name will have to be sufficient. Of course, all this may change with a revision of the analog descriptor format.

ADESC Digital control field

The digital control field is 6-bytes in length and also exhibits two format types:



As for the digital status case, there is support for up to 3 digital control actions related to a channel, corresponding to the digital status bits described above. If extended Bit# support for Bit#'s > 255 is used, there is support for one or two control actions. The dct values consist of two 4-bit fields. The most significant nibble of the byte is the pulse delay, if needed, and the least significant nibble gives the digital control type# in the range 0–15 described in the table above. All control actions listed in the table can be supported. For types 6 and 7 (and the hardware-driven equivalents 14 and 15, there is reference to a pair of bits used for separate control lines. Such a pair of control lines must be interfaced as a pair of adjacent bits in order that itemizing one bit of the pair implicitly defines the other. Thus, they may be bits 7 and 6, or they may be bits 1 and 0, but they cannot be bits 6 and 5, for example. Either bit of the pair can be entered in the database; the other is found by exclusive or of the Bit# with the constant "1." In the parameter page treatment of this facility of digital control relating to an analog channel, the state value of 0 or 1 is taken from the cursor location at the point of interrupt; if it was in the left half of the status display field, the state value is 0, otherwise 1. It is often referred to as the left/right state for this reason.

Let's have an example

Suppose there is a power supply which supplies current to a magnet and which has an on-off status bit and an "interlocks tripped" status bit, and it uses a pair of on-off control lines plus a "reset" control line which resets the "interlocks tripped" condition. How should the fields in the analog descriptor be programmed to support this device?

Let's not presume that the Bit#'s are small. We must therefore use the format type#1 described above. Suppose that the Bit#'s are as follows:

Bit#	Meaning
19F	On control pulse (0.5 sec hi-active pulse)
19E	Off control pulse (0.5 sec hi-active pulse)
19D	Reset control pulse (1.0 sec hi-active pulse)
1AF	On-off status (1=on)
1AE	Interlocks (0=tripped)

One possible solution would be the following:

The title text in the local database might be: `"---- OK BAD OFF ON"`

And the parameter page display might appear as follows:

```
"MVT102- OK ... ... ON  1013  A  "
```
or
```
"MVT102- ...BAD OFF...  0.622 A  "
```

Summary

The VME system software supports digital status and control with choices to accommodate a variety of hardware characteristics. The digital data can be memory-mapped bits, or it can be interfaced via 1553. Control bits can be set or cleared or toggled in a “dc” mode, or they can be pulsed hi or lo with a choice upon length of the pulse for software-controlled pulses. Hardware-controlled pulsing is also recognized and provided for. Pairs of bits can be used to provide two control lines that relate to a single status bit.

There is also provision for channel-related status and control bits for the convenience of displays such as the parameter page. Status field text can be used to show the state of status bits, and related control actions can be accepted with only simple setting commands issued from the display program; the details are kept in the local database.

It is hoped that this brief tutorial will serve as an introduction to the variety of digital status and control support provided by the VME systems. In addition, it may inspire host level user programs to devise ever-simpler means of dealing with digital I/O, found in real life to be rather more complicated than simply setting 1’s and 0’s.